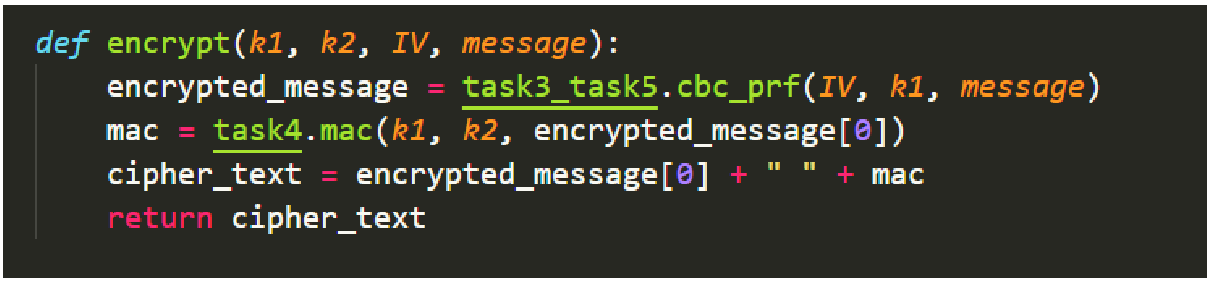
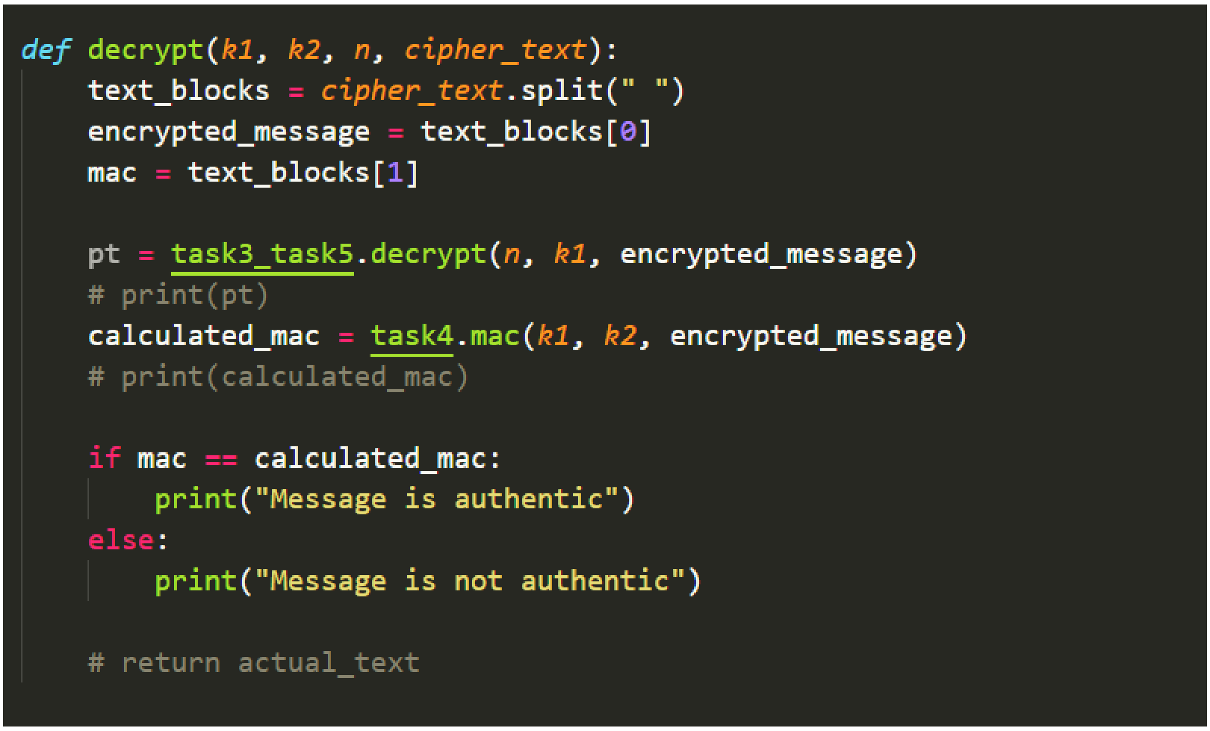
Task 5 CCA



```
def encrypt(k1, k2, IV, message):
    encrypted_message = task3_task5.cbc_prf(IV, k1, message)
    mac = task4.mac(k1, k2, encrypted_message[0])
    cipher_text = encrypted_message[0] + " " + mac
    return cipher_text
```

Encrypt()

Takes key1, key2, Initialization Vector and Message as input, it calls functions implemented in task3 and task4 to calculate encrypted text as well as the mac of the message to be sent. The resultant values are than appended and returned to the calling function.



```
def decrypt(k1, k2, n, cipher_text):
    text_blocks = cipher_text.split(" ")
    encrypted_message = text_blocks[0]
    mac = text_blocks[1]

    pt = task3_task5.decrypt(n, k1, encrypted_message)
    # print(pt)
    calculated_mac = task4.mac(k1, k2, encrypted_message)
    # print(calculated_mac)

    if mac == calculated_mac:
        print("Message is authentic")
    else:
        print("Message is not authentic")

    # return actual_text
```

Decrypt()

Takes key1, key2, IV and cipher_text as input. The function calculates the mac of the received cipher_text, if the mac matches the sent mac, the message is decrypted and a message regarding whether the message was authentic or not is printed on the screen.

Sample Input

Key1: 111010

Key2: 101010

IV: 100110

Message: 101010010101101101111

Sample Output

OG Text: 101010010101101101111

MAC: 010011

Message is Authentic